Introduksjon

I denne mappeoppgaven i faget Visualisering og Simulering, skal vi lage et individuelt visualisering av 3D terreng i en spillmotor. Det skal inkludere en kort video som forklarer hovedkoden, og ha et dokument som rapporterer framgangen og resultatet.

Det er verdt å påpeke at faglærer har nevnt at man kan jobbe i grupper, så lenge man refererer til hvem som har gjort hva. I denne oppgaven, satt 3 medstudenter seg ned og snakket om framgangsmåten til å få koden til å fungerer, enn å skrive en felles kode.

Formålet med å lage denne oppgaven er for å vise kunnskapen om metoder og verktøy for 3D visualisering og simulering. Deretter så er hovedmålet å sette opp og visualisere og simulere matematiske/fysiske modeller i et 3D-API. Ved å vise fram dette på programvaren vi har blitt undervist i før (OpenGL) har vi bedre kunnskap i å hente og videreutvikle kode vi har jobbet med. Dette viser også kunnskap i å løse og videreutvikle en kode som skal fungere tidligere, men får ny kode som oppdaterer data.

I obligatorisk innleveringsoppgave 3, var målet vårt å lage en simulering av en ball som sklir, ikke ruller, som beveger seg langs 4 trekanter ved hjelp av fysikk vi skriver inn manuelt og punkter vi har hentet fra den første obligatorisk oppgave. Ved å vise fram dette, kunne vi henvise oss til mappeoppgaven som skulle være i en større skala.

Mappeoppgaven krever at man bruker LAS eller SOSI filer og rendre ut en terreng. Videre skal man slippe en simulert ball vilkårlig sted i kartet og simulere nedbør og regn, ved hjelp av forenklet fysikk kalkulasjoner. Disse skal falle ned fra høy posisjon og se hvor de renner mot sluttpunktet. Hensikten med denne oppgaven er for å vise at man kan lage en visualisering av simulasjon ved bruk av Opengl sin 3D-motor. Ved å starte med å se over tidligere kode som fungerer, kan man gjenkjenne seg i

familiær kode som er skrevet av seg selv. Dette medfører til at mye kan erstattes med noe nytt, og fortsatt ha god oversikt over hva som har gått og hva som går galt.

Til tross for mye nytt materiale som må ta med i hensyn, så vil denne mappeoppgaven vise hva som kan gjøres med de enkelte stegene som er nevnt i oppgaveteksten. Selv om studenten har mye av koden og ideen bak oppgaven klart fra før, så er hovedspørsmålet fortsatt om hvordan kan vi lage/implementere et realistisk kart med baller som simulerer regn og følger newtons lover?

Metode

Første tingen vi måtte gjøre var å lage en kopi av oblig 3 eller å lage en kopi av en tidligere eksamensoppgave vi har gjort, for å gjenbruke det vi allerede har gjort fra før, og spare oss tid. For å kunne forsikre at det ikke skulle komme opp problemer framover, ble det brukt Github på grunn av dets historikk for kode. Vi skulle lagre en kopi av mappen og kopiere det til en ny github repository.

Etter å ha lagret klart en ny repository, skulle vi hente en fil vi kunne bruke til mappeoppgaven. Hovedmålet i denne var å bruke en LAS eller SOSI fil til oppgaven som leser punkter fra et geolokasjon i Norge eller andre deler av verden. Terratec forklarer “En laserpunktsky er et allsidig produkt som kan klassifiseres på flere ulike måter, for eksempel vil et vanlig produkt være en bakkeklassifisert punktsky som gir en nøyaktig terrengmodell.” (*Klassifisert Punktsky*, n.d.). Faglærer forklarer også på sin forelesningsnotat at “LAS-formatet er et filformat for geodata representert ved en punktsky. Det er et binært format.” (Nylund, 2022)

Dataen som studenten har hentet fra er fra en anbefalt side av faglærer, hoydedata.no. LAS filene kommer som et komprimert datasett, med navnet LAZ, og må bli håndtert via et verktøysett som heter LAStools via https://laszip.org/ eller ved å skrive inn egen kode. Det ble brukt det digitale verktøyet, grunnet til at mye av dette har blitt rettet opp og fått en revisjon av programmerere som har jobbet med denne filen.

XYZ formatet er en enkel forklaring av hvordan en kode skal leses inn. For hver linje, kommer det tall og posisjoner inn til koden, som skal forklare hvert punkt i .txt filen. Ved flere punkter, kan man forklare hvilke deler av XYZ som skal koble seg sammen og lage en flate langs bakken. Dette kan man heldigvis hente fra tidligere oppgaver som er gjort fra forrige semester, der studentene hadde 3D-programmering. Dette kurset gjorde også klart å få et interaktiv karakter til å bevege seg langs flatene ved bruk av barysentriske koordinater.

I hovedoppgaven så ble det nevnt at man skulle bruke GLDrawElements, en måte å skrive XYZ formatet på i en ikke trekant måte. Problemet som medførte denne framgangsmåten ble at vi ikke hadde fått dypt nok forståelse og opplæring i Delaunay triangulering. Delaunay Triangulering er en måte å vite hva som er de to største vinklene i en firkant, som gjør nabo informasjonen lettere å håndtere enn vanlig annet. Det ble godkjent å bruke GLDrawArrays av faglærer, som medførte til en lettere håndtering av de innleste filene fra LAS filen.

I oppgaven, etter å ha konstruert et terreng for oppgaven, ble det ikke vist på skjermen. Grunnen til at dette skjedde er at posisjonen til LAS filen er langt vekk fra origo i OpenGL, og må derfor settes inn til riktig posisjon. Etter det ble riktig offset, viste kartet seg visuelt over området. Oppgaven spurte dermed etter å sette klart et kameravisning av område, slik at man kunne se hvor hele kartet ligger med en gang. Shadere måtte også settes på plass, slik at vi kunne visualisere skygger, lys, og sette på eventuelle andre teksturer på selve kartet om man skulle ønske. Med dette nevnt, så var det i oppgave å sette på en ekvidistanse ved hjelp av barysentriske koordinater. Studenten gjorde framgangsmåten ved å skrive inn i en shader at for hver Z meter høyde oppover, så kom det en linje over hele X og Y akse på hele kartet. ‘

Etter at dette ble gjennomført, var det å begynne på simulering av nedbør og vassdrag. Mappeoppgaven vil ha en simulering av en eller flere baller, som følger newtons andre lov, der et legeme blir påvirket av en eller flere krefter, så vil det få en akselerasjon i den retningen kreftene virker. (*Newtons Lover – Store Norske Leksikon*, 2021)

Denne loven kan beskrives med formelen:

$$F = ma$$

Ballene skal også ha mulighet til å ha et vilkårlig startpunktet i render vinduet, der man trykker på en knapp og tilkaller ballen rett ved denne posisjonen. Grunnlaget til at man skal velge posisjonen er for å tilrettelegge at ballene ruller på terrenger og skal ha en kollisjonsrespons der ballene treffer hverandre og beveger seg vekk, basert på hvordan de treffes. Der ble det valgt å la ballen starte direkte fra kamera posisjonen er på, slik at man kan se ballen rulle uten å måtte bevege kameraet hele tiden.

Med tanke på at baller skal starte ved vilkårlig posisjon, så er det viktig å poengtere at noen genererte baller skal starte høyt over bakken for å simulere nedbør i mappeoppgaven. De små ballene som man genererer skal være små, men det skal være mange av dem, for å vise fram mye av hvor regnet posisjoneres og reiser fra et sted til et annet over et gitt område. Hvis de stopper på et felles sted, og det er for mange på et sted, så blir ballene dyttet fremover til et annet sted, lengre vekk fra første stopp. Hvis det kommer en kant som stopper nedover akselerasjonen til ballene, så må dette simuleres i tillegg. Mye forskjellige situasjoner som kan vises ved hjelp av disse små genererte ballene. Til tross for at det kanskje ikke er helt nøyaktig som virkeligheten, fordi vann siger seg ned gjennom jorden på grunn av tyngdekraften (*Kapillært Vann – Store Norske Leksikon*, 2022). På grunn av dette, blir det mer en simulasjon om hvis det skal være en ball som reiser over jorden, så vil ikke jorden eller annet suge seg til vann og gjøre at det blir borte fra simuleringen.

Den siste oppgaven var også å lage et simulering av regndråpenes posisjon fra når de treffer bakken og deretter for hvert tidssteg. Disse posisjonene som man skulle registrere skulle deretter lages til en kvadratisk spline kurve. Den sistnevnte oppgaven og frivillige oppgaver, samt annet som ikke har blitt nevnt, har ikke blitt vist eller beskrevet i dybde, på grunn av tidsmessige grunner. Siden dette var de siste oppgavene, ble det minst prioritert av mappeoppgaven. Det var mye annet som

måtte fungere før man kunne få til noe, og ville vært vanskelig å håndtere hvis de forrige oppgaven ikke ble gjort på en riktig måte først.

Resultater

I begynnelsen tok studenten i gang med å hente en LAS fil fra nettsiden hoydedata.no, der det var viktig å vise fram en bratt fjellside og hvordan ballene ruller. Filen som blir brukt i selve prosjektet er av Glitterholet. Mens dette var gjort,så kunne studenten hente fram tidligere prosjekt og lage en ny repository, for å fremvise en .git fil med ny informasjon om commit historikken. Grunnen er for å se om man har gjort arbeidet grundig og kan hente fram tidligere versjoner av arbeid man har gjort, som også kan fungere.

Framgangsmåten på å lage en LAS fil var som følger: Hentet filen fra et vilkårlig lokasjon, vente til det ble sendt en epost til den som forespurte dette, hente inn LAZ formatet og bruke den via LASTools. Dette førte til at filen ble skrevet til en .txt fil med koordinater i XYZ og lese antall linjer som er laget i denne oppgaven. Ved å legge inn denne nye XYZ koordinatene, var det viktig å fremvise at ballen kunne rulle på selve LAS .txt filen.

Selve filbehandlingen sin vei er som følger: Programmet starter, og går til renter vinduet. Rende Vinduet går videre til mainwindow og fører til renderwindow c++ filen. Renderwindow initialiserer filer og scener, som går til en egenlaget scene som heter physicsScene. Denne physicsScene inneholder alt vi skal bruke til oppgaven, som først og fremst henter en createObjects funksjon. I denne funksjonen, henter den en temp fil for visuelleobjekter og selve .txt filen.

```
LAZSurface* surface{nullptr};

   mObjects.push_back(surface = new
LAZSurface("../Mappeoppgave/Surface/GlitterholetShortened.txt",
QVector2D(600,300),*this, mShaderHandler->mShaderProgram[0],
QVector3D(-473213.f-1110/2, -6835647.f - 2110/2, -1734.f)));

   surface->setName("LasFile");
```

Ved å vise et avsnitt av koden, som er forkortet til å videre forklare framgangsmåten og hvordan det tenkes å framføres.

Alle filer er et visuelt objekt, der man plasserer filen i en spesifikk område av minnet og kan framkalle dette senere. I filen, så henter man en peker til et annen del av koden, og plasserer det i konstruktøren til filen. I konstruktøren er det satt opp slik at man skriver inn navnet der .txt LAS filen ligger, i en string format. Deretter når studenten komprimerer filen, triangulerer studenten selve området som tas i bruk, ved å sette en grid størrelse til filen, for å ta med det som er det man mener er viktigst i koden. I studentens kode spesifiserer man hvilken scene man bruker, som er denne spesifikke scenen, og hvilken shaderprogram denne koden skal følge. Til slutt, så setter man av på riktig offset, siden vi ønsker at koden skal være i origo av scenen vår og ikke i langt vekk, der den originalt ble hentet fra i xyz koordinatene. Deretter beskriver du at denne peker filen skal ha et navn som er passelig til koden, og da er man ferdig til å lage selve koden.

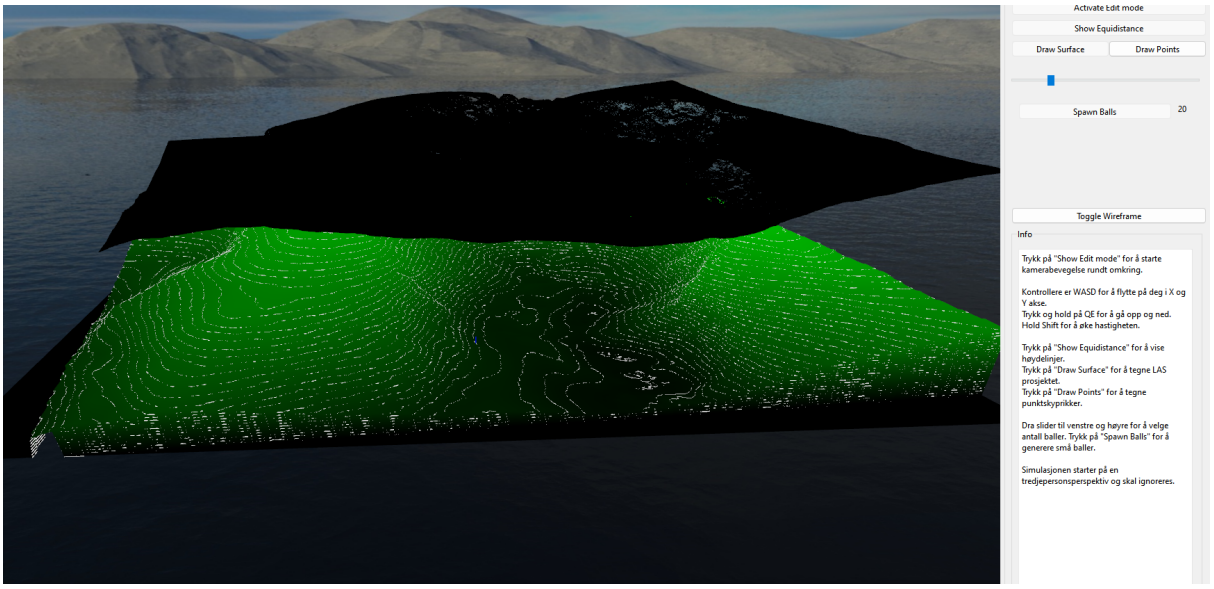
Når selve koden går inn til å bli lest opp av LAZSurface, går .txt filen til construct og readfile funksjonene. Disse leser inn all data man har fremhevet tidligere under datasettet og blir laget og lest av maskinen for å visualisere terrenget.

For å se resultatet, fikk studenten i oppgave på å tilpasse koordinatsystemet i rendervinduet og kameraposisjonen. Målet var å la individet som starter prosjektet til å til å sette i gang med en riktig kameraposisjon. Dessverre, grunnet til at det ble en usikker framgangsmåte av studenten, starter prosjektet med et låst kamera til en player, der man må trykke på “Show Edit mode” og bevege kamera over et visst punkt til å se LAS fil terrenget visualisert.

Høydekurvene som ble implementert var ekvidistanse som var på mer enn ønsket høyde, på grunnlaget av at høye fjell gjorde det mye vanskeligere og mer rotete å se på selve koden. Prosedyren for dette er veldig likt framgangsmåte som txt filen. Forskjellen er at ekvidistansen skal bruke surface pekeren i tillegg.

```
   mObjects.push_back(equidistance =
surface->constructEquidistance());
   equidistance->setName("Equidistance");
```

Siden vi skal ha en ekvidistanse på samme overflate, kan vi bruke samme peker som tidligere og henvise oss til at linjene konstrueres rundt et område som er hentet i barysentriske koordinatene. For hvert Z høyde opp, så vil det lage en ekvidistanse i hele X og Y posisjon. Grunnet til at det å gå igjennom hele koden vil medføre færre sider å forklare alt, så vil denne avvikles for dypere forklaring og heller vise fram et bilde av resultatet.



For oppgaven av simulering av nedbør og vassdrag, så kom det opp flere problemer enn antatt. Det å lage flere baller som skulle settes på et vilkårlig område over bakken var mulig, men uheldigvis vil ikke koden slippe nedbørs ballene rett ovenfor selve terrenget, og genererer seg heller utenfor området. Når regndråpene treffer bakken, spretter de uten noe form for friksjonskraften som ødelegger for selve ballene. Hvis det er sånn at disse små genererte ballene er for lenge ute, så kan minne gå utenfor vector antallet og blir derfor en kræsj i systemet. Vi kan heller ikke øke hastigheten for mye, for ballene vil gå gjennom bakken og falle videre ned. Dette har medført til at oppgaven viser selve videoen kun en gang. Oppgave 3.1 og 3.3 ble ikke gjennomført, grunnet til dårlig planlegging av studenten.

Diskusjon

Hva medfører resultatet vi så over til helheten av mappeoppgaven? Det som viser seg er at for å kunne få et bra forståelse av faget, så finner vi mer ut av hva vi trenger mer tid og dypere utdanning til fysikk. På grunnlaget av faglærere gav studentene halve semesteret til å kun fokusere på et fag, skapte det egentlig mer negativ læring. Det var mye mer stressende enn et vanlig semester med 2 andre fag i tillegg.

Det er mye å snakke om i denne oppgaven, som har sine sterke sider og sine svake sider. Styrkene ved å jobbe med denne oppgaven var å kunne snakke sammen om ideer om hvordan beste løsning muligheten var, enn å måtte gruble matematiske formler i lengre periode. Svakhetene ved å jobbe på denne mappeoppgaven var det å få for lite utdanning og læring av selve faget og gjorde selve mappeoppgaven både enklere og vanskeligere. Det var beleilig å få muligheten til å avvikle oppgaver som vi ikke hadde nok informasjon om og prioritere det som var nødvendig. Men det skapte også et ønske om å gjøre det på en ordentlig måte og vise det fram til en framtidig showreel man selv ønsket å ha. Samtidig det å kutte ned semesteret til et halvt semester, betydde også at hvis det er noe man ikke klarer å forstå, så har man halvparten av tiden til å fremvise at studenten har noe godt forståelse.

Oppgaver som studenten ikke fikk mulighet til å gjennomgå vil være et noe usvarte spørsmål og ønsket om å få mer tid til å videre gå inn i emnet til å undersøke hva som kunne bli optimalisert for å få en flom scenario på glitterholet. Om dette hadde vært en flom situasjon, i lik grad som 2015 filmen bølgen, der det er en flodbølge i Geirangerfjorden.

For å svare på spørsmålet som ble spurt i tidligere i oppgaven: hvordan kan vi lage/implementere et realistisk kart med baller som simulerer regn og følger newtons lover? Det er vist at det er mye som skal til for å få ting til å sette seg ordentlig og fungere, men om man bruker god kodestruktur og forstår materialet man jobber med, så har man muligheten til å forbedre arbeidet.

Referanser

kapillært vann – Store norske leksikon. (2022, April 1). Store norske leksikon. Retrieved October 5, 2022, from https://snl.no/kapill%C3%A6rt_vann

Newtons lover – Store norske leksikon. (2021, September 2). Store norske leksikon. Retrieved October 5, 2022, from https://snl.no/Newtons_lover

Nylund, D. (2022, September 29). *MAT301 Matematikk III - Visualisering og Simulering forelesningsnotater og oppgaver*. Retrieved Oktober 05, 2022, from https://drive.google.com/file/d/1VGRtC5Nn_pTVbmwZLA2kHxEhtRQ2LfGo/view

Terratec AS. (n.d.). *Klassifisert punktsky*. Terratec AS. Retrieved October 5, 2022, from https://terratec.no/tjenester/kartleggingstjenester/klassifisert-punktsky/